Report | HW 2

CS663 - Digital Image Processing

Omkar Shirpure (22B0910)
Suryansh Patidar (22B1036)
Aditya Saran (22B1839)

Contents

1 Q1

$$I_{\text{noisy}}(x, y) = I(x, y) + \eta(x, y)$$

where $\eta \sim \mathcal{N}(0, \sigma^2)$.

The PDF of η, denoted p_η, is:

$$p_\eta(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

For the PDF of the resulting noisy image, we know that for two independent random variables X and Y, if $Z = X + Y$, then the PDF of Z is given by:

$$p(Z) = \int_{-\infty}^{\infty} p_X(x) p_Y(z - x)\, dx$$

which is the convolution of the PDFs of X and Y.

To prove that if X and Y are independent random variables, then the probability density function (pdf) of $Z = X + Y$ is given by the convolution of the pdfs of X and Y, we can proceed with the following steps:
Let $f_X(x)$ and $f_Y(y)$ denote the probability density functions of the independent random variables X and Y, respectively.
We need to show that the pdf of $Z = X + Y$, denoted as $f_Z(z)$, is the convolution of $f_X(x)$ and $f_Y(y)$.
The pdf of Z is given by:

$$f_Z(z) = \frac{d}{dz} P(Z \leq z) = \frac{d}{dz} P(X + Y \leq z)$$

Since X and Y are independent, the joint probability distribution can be expressed as the product of their individual distributions:

$$f_Z(z) = \frac{d}{dz} \int_{-\infty}^{\infty} P(X \leq z - y) f_Y(y)\, dy$$

Taking the derivative with respect to z:

$$f_Z(z) = \int_{-\infty}^{\infty} \frac{d}{dz} P(X \leq z - y) f_Y(y)\, dy$$

Since the derivative of the cumulative distribution function $P(X \leq z - y)$ is the pdf $f_X(z - y)$, we get:

$$f_Z(z) = \int_{-\infty}^{\infty} f_X(z - y) f_Y(y)\, dy$$

Thus, the pdf of $Z = X + Y$ is the convolution of the pdfs of X and Y, which is written as:

$$f_Z(z) = (f_X * f_Y)(z) = \int_{-\infty}^{\infty} f_X(z - y) f_Y(y)\, dy$$

So our PDF of the noisy image, p_{noisy}, is the sum of the PDF of the original image I and noise η, so the PDF of the resulting image will be:

$$p_{\text{noisy}}(x) = \int_{-\infty}^{\infty} p_I(y) p_\eta(x - y)\, dy = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{\infty} p_I(y) e^{-\frac{(x-y)^2}{2\sigma^2}}\, dy$$

Since the noise follows a Gaussian distribution, it acts as a Gaussian filter over our image, and thus it resembles Gaussian blurring (or Gaussian smoothing).

Now in the second case, if the noise is uniformly distributed in the range $[-r, +r]$, the PDF of the noise becomes:

$$p_{\eta}^{'}(x) = \begin{cases} \frac{1}{2r} & \text{if } -r \leq x \leq r \\ 0 & \text{otherwise} \end{cases}$$

Here too the expression of the PDF of resulting noisy image will be convolution of our Original image and the noise

$$p_{\text{noisy}}(x) = \int_{-\infty}^{\infty} p_I(y) p_{\eta}^{'}(x-y)\, dy = \begin{cases} \frac{1}{2r} \int_{(x-r)}^{(x+r)} p_I(y)\, dy & \text{if } -r \leq x-y \leq r \\ 0 & \text{otherwise} \end{cases}$$

Since $p_{\eta}^{'}(x)$ is a uniform distribution, this resembles the box filtering operation in image processing, where the original image intensities are convolved with a uniform (box-shaped) kernel, leading to smoothing over a defined range of values

2 Q2

Let the 1D image be represented by a vector

$$f = \{f_0, f_1, \ldots, f_n\}$$

and a 1D convolution mask

$$w = \{w_0, w_1, \ldots, w_6\}$$

We will rotate w by $180°$ because we are performing convolution

The convolution of f with w at i^{th} point is

$$(f * w)_i = w_0 f_i + w_1 f_{i+1} + \cdots + w_6 f_{i+6}$$

This convolution can be expressed in matrix form as

$$f_{\text{conv}} = W \cdot f$$

where W is a **(n+1)** $\times$ **(n+1)** matrix that represents the convolution kernel and f is the image vector. The matrix W for a convolution mask of size 7 can be written as

$$W = \begin{pmatrix} w_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ldots \\ w_1 & w_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ldots & \\ w_2 & w_1 & w_0 & 0 & 0 & 0 & 0 & 0 & 0\ldots & \\ w_3 & w_2 & w_1 & w_0 & 0 & 0 & 0 & 0 & 0\ldots & \\ w_4 & w_3 & w_2 & w_1 & w_0 & 0 & 0 & 0 & 0\ldots & \\ w_5 & w_4 & w_3 & w_2 & w_1 & w_0 & 0 & 0 & 0\ldots & \\ w_6 & w_5 & w_4 & w_3 & w_2 & w_1 & w_0 & 0 & 0\ldots & \\ & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

and

$$\mathbf{f}_{\text{conv}} = \begin{pmatrix} w_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ldots \\ w_1 & w_0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\ldots & \\ w_2 & w_1 & w_0 & 0 & 0 & 0 & 0 & 0 & 0\ldots & \\ w_3 & w_2 & w_1 & w_0 & 0 & 0 & 0 & 0 & 0\ldots & \\ w_4 & w_3 & w_2 & w_1 & w_0 & 0 & 0 & 0 & 0\ldots & \\ w_5 & w_4 & w_3 & w_2 & w_1 & w_0 & 0 & 0 & 0\ldots & \\ w_6 & w_5 & w_4 & w_3 & w_2 & w_1 & w_0 & 0 & 0\ldots & \\ & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix} \cdot \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ \vdots \\ f_n \end{pmatrix}$$

2.1 Properties of the Matrix W

The matrix W has the following properties:

- **Toeplitz Structure**: Each descending diagonal from left to right is constant, meaning that the same weights from the convolution mask are shifted across rows.
- **Band-Diagonal**: The matrix is sparse and contains non-zero elements only in a band of diagonals, reflecting the local neighborhood convolution operation.
- **Shift-Invariance**: The structure of the matrix encodes the shift-invariant nature of convolution, i.e., the same weights are applied to different positions in the input vector.

2.2 Potential Application of Matrix-Based Convolution

One potential application of this matrix-based construction is in filtering operations such as:

- **Image Smoothing or Sharpening**: Applying convolution with Gaussian, Sobel, or Laplacian filters can be efficiently expressed using matrix multiplication, especially in cases where the filter remains constant but needs to be applied repeatedly to different inputs.
- **Convolutional Neural Networks (CNNs)**: In CNNs, the convolution operation is central, and expressing convolutions as matrix multiplications is useful for backpropagation and optimization.

This matrix-based approach can simplify the implementation of certain types of linear transformations. It also makes it easier to perform large-scale convolution operations.

3 Q3

3.1 Laplacian mask with -8 in the center is

$$\nabla^2 = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

A mask is separable if it can be written as an outer product of two 1D masks (or vectors).

Let us assume that this Laplacian mask can be written as an outer product of two filters:

$$\nabla^2 = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \begin{pmatrix} d & e & f \end{pmatrix} = \begin{pmatrix} ad & ae & af \\ bd & be & bf \\ cd & ce & cf \end{pmatrix}$$

By comparison: - $ad = 1$, $ae = 1$, $af = 1$ - $bd = 1$, $be = -8$, $bf = 1$ - $cd = 1$, $ce = 1$, $cf = 1$

We can see that: - $ad = ae = af = bd = bf = cd = ce = cf = 1$ So, $d = e = f$.

$$d = e = f \quad \Rightarrow \quad d = e = f = \frac{1}{a}$$

$$be = (a)(\frac{1}{a}) = 1 \neq -8$$

So, this Laplacian with -8 in the centre is **not a separable filter**.

3.2 Laplacian mask with -4 in the center is

$$L = \nabla^2 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

This came from the equation

$$L = \nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

I can split it in two parts such that

$$L = (f(x+1, y) - 2f(x, y) + f(x-1, y)) + (f(x, y+1) - 2f(x, y) + f(x, y-1))$$

And to show if this Laplacian mask can entirely be implemented with 1D masks, I need to prove,

$$I * L = I * K + I * M$$

where K and M are 1D-masks
So from above equation, i can write

$$L = (f(x+1, y) - 2f(x, y) + f(x-1, y)) + (f(x, y+1) - 2f(x, y) + f(x, y-1)) = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 & -2 & 1 \end{pmatrix}$$

where **K** = $\begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}$ and **M** = $\begin{pmatrix} 1 & -2 & 1 \end{pmatrix}$ So i can write,

$$I * L = I * (K + M)$$

and due to distrbutivity (linear) property of Convolution operator i can write

$$I * (K + M) = I * K + I * M$$

Where K and M have been defined and both are 1D. So hence we can say that Laplacian mask with a −4 in the center can be implemented entirely using 1D convolutions.

Proof of the distrbutivity property of Convolution Operator

$$\begin{aligned}(x_1 * (x_2 + x_3))(t) &= \int_{-\infty}^{\infty} x_1(\tau)(x_2(t-\tau) + x_3(t-\tau))\, d\tau \\ &= \int_{-\infty}^{\infty} x_1(\tau)x_2(t-\tau)\, d\tau + \int_{-\infty}^{\infty} x_1(\tau)x_3(t-\tau)\, d\tau \\ &= (x_1 * x_2)(t) + (x_1 * x_3)(t)\end{aligned}$$

4 Q4

4.1 What happens with Addition

Taking the below as an example for 1D image:

$$\boxed{0 \quad 0 \quad 0 \quad 0 \quad 10 \quad 10 \quad 10 \quad 0 \quad 0 \quad 0 \quad 0}$$

The expression for the second derivative in 1D image is

$$\nabla^2 I(x) = I(x+1) - 2I(x) + I(x-1)$$

So, the second derivative for the above image is:

$$\boxed{0 \quad 0 \quad 10 \quad -10 \quad 0 \quad -10 \quad 10 \quad 0 \quad 0}$$

We can see that the 2nd derivative is changing sign at 2 places, first from 3rd to 4th index and second from 6th to 7th index. Therefore there are 2 edges. Now we will look at second image after adding $\alpha\nabla^2 I(x,y)$ to the intensity values in the original image

Taking $\alpha = 0.5$, final intensity values (update rule: $I(x) \leftarrow I(x) + 0.5 \times \nabla^2 I(x)$) will be:

$$\boxed{0 \quad 0 \quad 0 \quad 5 \quad 5 \quad 10 \quad 5 \quad 5 \quad 0 \quad 0 \quad 0}$$

The second derivative for this image is

$$\boxed{0 \quad 5 \quad -5 \quad 5 \quad -10 \quad 5 \quad 0 \quad 0}$$

We see that wherever there are zero crossings of our resultant image, the step is decreased as compared to step in our original image. See the 3rd and 4th index in new image and also the 6th and 7th in new image. The step for zero crossing has reduced from 20 to 10. Hence the edges are blurred.
The Laplacian tends to smooth out rapid changes in intensity because it highlights areas where there are large changes in slope (i.e., edges). Adding $\alpha\nabla^2 I(x)$ to the original signal $I(x)$ effectively reduces these rapid changes, leading to blurring of the signal.

For example, if we have a sharp edge in $I(x)$, $\nabla^2 I(x)$ will produce large values at the edge, and adding $\alpha\nabla^2 I(x)$ will diminish the intensity gradient at that edge, thus blurring the edge just like in above example.

When the operation $I(x,y) \leftarrow I(x,y) + \alpha\nabla^2 I(x,y)$ is iterated for many steps:

- **Blurring over time**: The image will progressively become smoother. Each iteration reduces the gradient at high-intensity areas (such as edges), causing the entire image to blur over time.
- **Convergence to a constant image**: If this process is continued indefinitely, the image will eventually converge to a constant intensity value everywhere (i.e., a fully blurred image where all gradients are zero). This is because the Laplacian will smooth out all variations in the image until it becomes uniform.

4.2 What happens with subtraction?

Taking same example, but now we perform $I(x) \leftarrow I(x) - 0.5 \times \nabla^2 I(x)$)
We will get our new image as,

$$\boxed{0 \quad 0 \quad 0 \quad -5 \quad 15 \quad 10 \quad 15 \quad -5 \quad 0 \quad 0 \quad 0}$$

The second derivative for this image is

$$\boxed{0 \quad -5 \quad 25 \quad -25 \quad -10 \quad -15 \quad 5 \quad 5 \quad 0}$$

We see that wherever there are zero crossings of our resultant image, the step is decreased as compared to step in our original image. See the 3rd and 4th index in new image. The step for zero crossing has increased from 20 to 50. Hence the edges are sharpened.

Hence for many iterations, the effect will be the opposite to that observed when we were adding the $\alpha\nabla^2 I(x, y)$ term:

- **Sharpening of edges**: Since $\nabla^2 I(x, y)$ is large where the image has high curvature (e.g., at edges), subtracting $\alpha\nabla^2 I(x, y)$ will enhance these features, effectively sharpening the edges.
- **Instability over many iterations**: If this operation is continued for many iterations, the image will become unstable, with the sharp features becoming more pronounced to the point that noise or small variations could become exaggerated. This can result in edge amplification or even the creation of spurious artifacts, potentially leading to chaotic patterns in the image.

5 Q5

5.1 Zero-Mean Gaussian Filter

The Gaussian filter in 1D has the form:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

We want to filter the image $I(x) = cx + d$ with this Gaussian filter, which is equivalent to computing the convolution:

$$J(x) = (I * G)(x) = \int_{-\infty}^{\infty} I(t)G(x-t)\,dt$$

Substitute $I(t) = ct + d$ and $G(x-t) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-t)^2}{2\sigma^2}}$:

$$J(x) = \int_{-\infty}^{\infty} (ct+d) \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-t)^2}{2\sigma^2}}\,dt$$

This integral can be separated into two parts:

$$J(x) = c\int_{-\infty}^{\infty} t \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-t)^2}{2\sigma^2}}\,dt + d\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-t)^2}{2\sigma^2}}\,dt$$

The second term is simply the convolution of a constant d with a Gaussian, which gives back d, since the Gaussian is normalized:

$$d\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-t)^2}{2\sigma^2}}\,dt = d$$

The first term involves the convolution of a linear function t with a Gaussian. For a linear ramp function, this convolution yields the same ramp function because the Gaussian only smooths the data, but a linear function remains unchanged in slope:

$$c\int_{-\infty}^{\infty} t \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-t)^2}{2\sigma^2}}\,dt = cx$$

Thus, the final expression for $J(x)$ after filtering the ramp $I(x)$ with a Gaussian is:

$$J(x) = cx + d$$

5.2 Bilateral Filter

The bilateral filter is a non-linear filter that considers both spatial proximity and intensity similarity. Its general form in 1D is:

$$J(x) = \frac{1}{W(x)} \int_{-\infty}^{\infty} I(t) e^{-\frac{(x-t)^2}{2\sigma_s^2}} e^{-\frac{(I(x)-I(t))^2}{2\sigma_r^2}}\,dt$$

where: - σ_s controls the spatial Gaussian kernel. - σ_r controls the range (intensity) Gaussian kernel. - $W(x)$ is a normalization factor:

$$W(x) = \int_{-\infty}^{\infty} e^{-\frac{(x-t)^2}{2\sigma_s^2}} e^{-\frac{(I(x)-I(t))^2}{2\sigma_r^2}} \, dt$$

For the ramp image $I(x) = cx + d$, we note that the intensity difference $I(x) - I(t) = c(x - t)$. Thus, the Gaussian in intensity becomes:

$$e^{-\frac{(I(x)-I(t))^2}{2\sigma_r^2}} = e^{-\frac{c^2(x-t)^2}{2\sigma_r^2}}$$

The bilateral filter thus simplifies to:

$$J(x) = \frac{1}{W(x)} \int_{-\infty}^{\infty} (ct + d) e^{-\frac{(x-t)^2}{2\sigma_s^2}} e^{-\frac{c^2(x-t)^2}{2\sigma_r^2}} \, dt$$

The product of the two Gaussian terms becomes a single Gaussian term with a combined variance:

$$e^{-\frac{(x-t)^2}{2\sigma_{\text{combined}}^2}} \quad \text{where} \quad \sigma_{\text{combined}}^2 = \frac{\sigma_s^2 \sigma_r^2}{\sigma_s^2 + c^2 \sigma_r^2}$$

Thus, the bilateral filter effectively behaves like a Gaussian filter with a modified standard deviation σ_{combined}, and the result of applying the bilateral filter to the ramp image is:

$$J(x) = cx + d$$

Conclusion

In both cases (Gaussian and bilateral filtering), the resulting image is the same as the original image $I(x) = cx + d$. This happens because the ramp function is a linear function, and both filters respect the linearity of the image. The ramp's structure remains unchanged under both filters.

6 Q6

We are given the following transformations:

$$u = xcos\theta - ysin\theta \tag{1}$$

$$v = xsin\theta + ycos\theta \tag{2}$$

Similarly, the inverse transformations are:

$$x = ucos\theta + vsin\theta \tag{3}$$

$$y = -usin\theta + vcos\theta \tag{4}$$

We are required to prove that:

$$I_{xx} + I_{yy} = I_{uu} + I_{vv} \tag{5}$$

Let's first calculate $I_{uu} + I_{vv}$ as follows:

$$I_{uu} + I_{vv} = \frac{\partial^2}{\partial u^2} + \frac{\partial^2}{\partial v^2} \tag{6}$$

$$= \frac{\partial}{\partial u}\left(\frac{\partial x}{\partial u}.\frac{\partial}{\partial x} + \frac{\partial y}{\partial u}.\frac{\partial}{\partial y}\right) + \frac{\partial}{\partial v}\left(\frac{\partial x}{\partial v}.\frac{\partial}{\partial x} + \frac{\partial y}{\partial v}.\frac{\partial}{\partial y}\right) \tag{7}$$

$$= \frac{\partial}{\partial u}\left(cos\theta\frac{\partial}{\partial x} - sin\theta\frac{\partial}{\partial y}\right) + \frac{\partial}{\partial v}\left(sin\theta\frac{\partial}{\partial x} + cos\theta\frac{\partial}{\partial y}\right) \tag{8}$$

$$= \left(cos\theta\frac{\partial}{\partial x} - sin\theta\frac{\partial}{\partial y}\right)^2 + \left(sin\theta\frac{\partial}{\partial x} + cos\theta\frac{\partial}{\partial y}\right)^2 \tag{9}$$

$$= cos^2\theta\frac{\partial^2}{\partial x^2} + sin^2\theta\frac{\partial^2}{\partial y^2} + sin^2\theta\frac{\partial^2}{\partial x^2} + cos^2\theta\frac{\partial^2}{\partial y^2} \tag{10}$$

$$= I_{xx} + I_{yy} \tag{11}$$

Thus, we have proven that:

$$I_{xx} + I_{yy} = I_{uu} + I_{vv} \tag{12}$$

Hence Proved.

Directional Derivatives

The First directional derivative of I(x,y) in the direction v is given by $\nabla I(x,y).\mathbf{v}$

Thus, Second directional derivative of I(x,y) in the direction v becomes $\nabla(\nabla I(x,y).\mathbf{v}).\mathbf{v}$

Now, for the second directional derivative of I(x,y) in the direction of gradient vector, we calculate:

$$\nabla\left(\nabla I(x,y).\left(\frac{I_x\hat{i} + I_y\hat{j}}{\sqrt{I_x^2 + I_y^2}}\right)\right).\left(\frac{I_x\hat{i} + I_y\hat{j}}{\sqrt{I_x^2 + I_y^2}}\right) \tag{13}$$

$$= \nabla\left((I_x\hat{i} + I_y\hat{j}).\left(\frac{I_x\hat{i} + I_y\hat{j}}{\sqrt{I_x^2 + I_y^2}}\right)\right).\left(\frac{I_x\hat{i} + I_y\hat{j}}{\sqrt{I_x^2 + I_y^2}}\right) \tag{14}$$

$$= \nabla \left(\sqrt{I_x^2 + I_y^2} \right) . \left(\frac{I_x \hat{i} + I_y \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) \tag{15}$$

$$= \left(\frac{I_x I_{xx} + I_y I_{xy}}{\sqrt{I_x^2 + I_y^2}} \hat{i} + \frac{I_x I_{xy} + I_y I_{yy}}{\sqrt{I_x^2 + I_y^2}} \hat{j} \right) . \left(\frac{I_x \hat{i} + I_y \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) \tag{16}$$

$$= \frac{I_x^2 I_{xx} + 2 I_x I_y I_{xy} + I_y^2 I_{yy}}{I_x^2 + I_y^2} \tag{17}$$

The second directional derivative in the direction perpendicular to the gradient vector is given by:

$$\nabla \left(\nabla I(x,y) . \left(\frac{-I_y \hat{i} + I_x \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) \right) . \left(\frac{-I_y \hat{i} + I_x \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) \tag{18}$$

$$= \nabla \left((I_x \hat{i} + I_y \hat{j}) . \left(\frac{-I_y \hat{i} + I_x \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) \right) . \left(\frac{-I_y \hat{i} + I_x \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) \tag{19}$$

$$= \nabla (0) . \left(\frac{-I_y \hat{i} + I_x \hat{j}}{\sqrt{I_x^2 + I_y^2}} \right) = 0 \tag{20}$$

7 Q7

7.1 Barbara Image: Noise ($\sigma = 5$)

(a) Original Image

(b) Noisy Image (Gau. noise with $\sigma = 5$)

(c) Filtered Image ($\sigma_s = 2, \sigma_r = 2$)

(d) Filtered Image ($\sigma_s = 0.1, \sigma_r = 0.1$)

(e) Filtered Image ($\sigma_s = 3, \sigma_r = 15$)

Figure 1: Barbara image with noise ($\sigma = 5$) and noise removal with given parameters of Bilateral Filter

7.2 Barbara Image: Noise ($\sigma = 10$)

(a) Original Image

(b) Noisy Image (Gau. noise with $\sigma = 10$)

(c) Filtered Image ($\sigma_s = 2, \sigma_r = 2$)

(d) Filtered Image ($\sigma_s = 0.1, \sigma_r = 0.1$)

(e) Filtered Image ($\sigma_s = 3, \sigma_r = 15$)

Figure 2: Barbara image with noise ($\sigma = 10$) and noise removal with given parameters of Bilateral Filter

7.3 Kodak Image: Noise ($\sigma = 5$)

(a) Original Image

(b) Noisy Image (Gau. noise with $\sigma = 5$)

(c) Filtered Image ($\sigma_s = 2, \sigma_r = 2$)

(d) Filtered Image ($\sigma_s = 0.1, \sigma_r = 0.1$)

(e) Filtered Image ($\sigma_s = 3, \sigma_r = 15$)

Figure 3: Kodak image with noise ($\sigma = 5$) and noise removal with given parameters of Bilateral Filter

7.4 Kodak Image: Noise ($\sigma = 10$)

(a) Original Image

(b) Noisy Image (Gau. noise with $\sigma = 10$)

(c) Filtered Image ($\sigma_s = 2, \sigma_r = 2$)

(d) Filtered Image ($\sigma_s = 0.1, \sigma_r = 0.1$)

(e) Filtered Image ($\sigma_s = 3, \sigma_r = 15$)

Figure 4: Kodak image with noise ($\sigma = 10$) and noise removal with given parameters of Bilateral Filter

7.5 Comparisons of the results

7.5.1 Barbara Image: Noise ($\sigma = 5$)

- ($\sigma_s = 2$, $\sigma_r = 2$) : This configuration will moderately smooth the image both spatially and in intensity. There is visible noise reduction with preservation of edge details. Also, very little fine details have become smoothed out.
- ($\sigma_s = 0.1$, $\sigma_r = 0.1$) : A small σ_s and σ_r means the changes will be very localised with similar intensity values. This has lead to very less o no filtering effect, slight the noise is mostly visible, and has very little smoothing (not much different from the previous one, but we'll see the difference when more noise is introduced).
- ($\sigma_s = 3$, $\sigma_r = 15$) : A larger σ_s and σ_r has resulted in strong spatial and intensity smoothing.This significantly reduces the noise, but at the cost of blurring finer details and edges. With σ_r being large, more neighboring pixels are considered, causing aggressive denoising. However, this over-smoothing has blurred important details, such as facial features or the texture of the background (e.g., the eyes or netting behind in (e)).

7.5.2 Barbara Image: Noise ($\sigma = 10$)

All the cases would be mostly similar in terms of magnitude of noise-reduction, edge-smoothing but as this contains more noise we observe magnified differences in individual parameters.

- ($\sigma_s = 2$, $\sigma_r = 2$) : Similar to the previous case, this has reduced noise to somewhat extent, preserved edges significantly, but some noise still persists.
- ($\sigma_s = 0.1$, $\sigma_r = 0.1$) : The image still appears quite noisy, with but seems more contrasty than previous case, as changes are impacted by more localised pixels.
- ($\sigma_s = 3$, $\sigma_r = 15$) : This has drastically reduced the noise, but as we can see, some edges and fine details are smeared/blurred out.

7.5.3 Kodak Image: Noise ($\sigma = 5$)

- ($\sigma_s = 2$, $\sigma_r = 2$) and ($\sigma_s = 0.1$, $\sigma_r = 0.1$) : The image seem mostly similar for lesser noise with a balance between noise reduction and edges-preservation.
- ($\sigma_s = 3$, $\sigma_r = 15$) : This configuration has reduced most of the noise, but as we can see, very edges and fine details (e.g. near balcony, design details above doors) are smeared/blurred out.

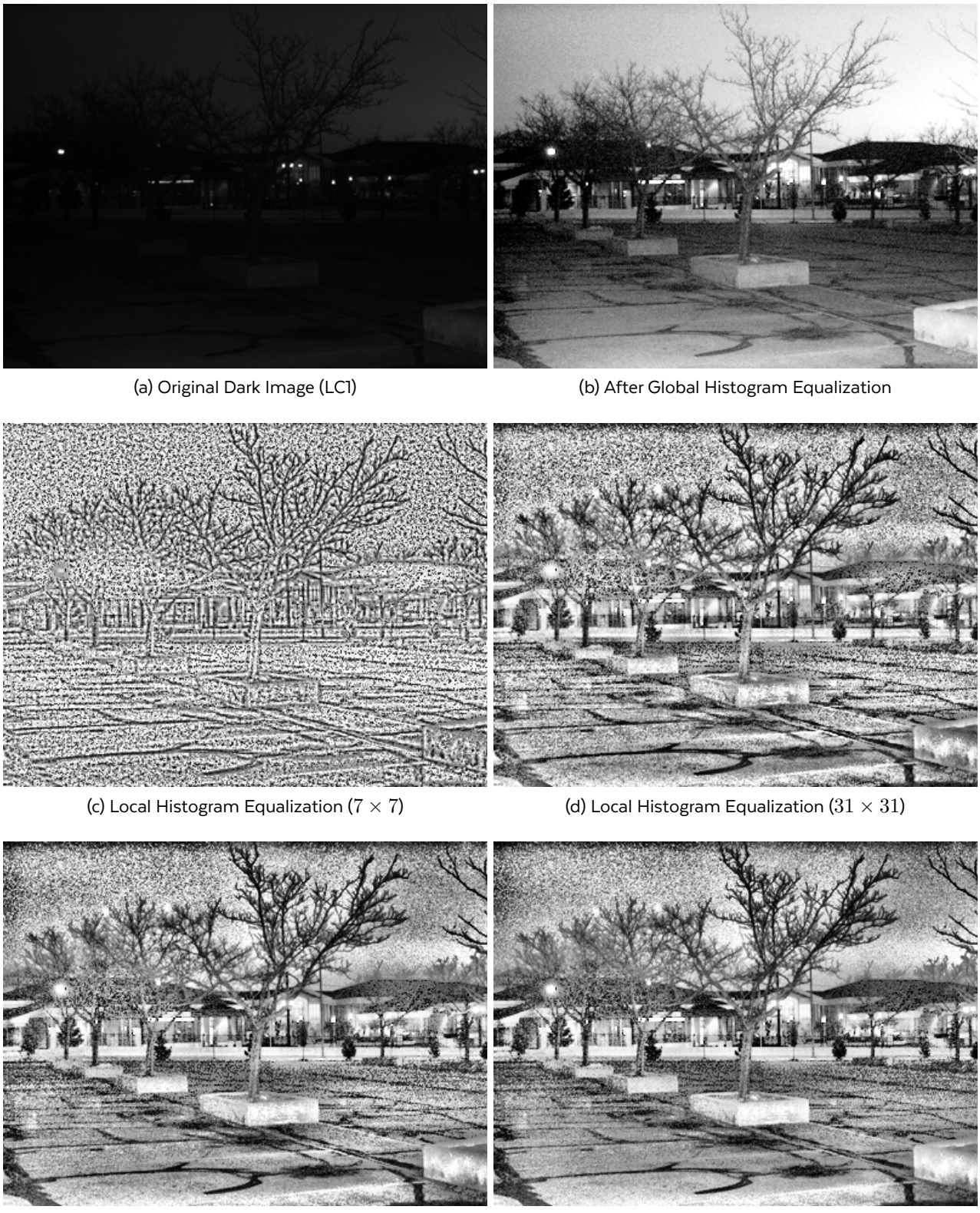
7.5.4 Kodak Image: Noise ($\sigma = 10$)

All the cases would be mostly similar in terms of magnitude of noise-reduction, edge-smoothing but as this contains more noise we observe magnified differences in individual parameters.

- ($\sigma_s = 2$, $\sigma_r = 2$) : Noise is significantly reduced along with preservation of edges.
- ($\sigma_s = 0.1$, $\sigma_r = 0.1$) : Bit noisy than previous result, but still good and maintains balance betwwen noise reduction and preserving details.
- ($\sigma_s = 3$, $\sigma_r = 15$) : This has drastically reduced the noise, but as we can see, some edges and fine details are smeared/blurred out.

8 Q8

8.1 Histogram Equalization Results on LC1



(a) Original Dark Image (LC1)

(b) After Global Histogram Equalization

(c) Local Histogram Equalization (7×7)

(d) Local Histogram Equalization (31×31)

(e) Local Histogram Equalization (51×51)

(f) Local Histogram Equalization (71×71)

Figure 5: LC1 image with global and local histogram equalization using given window sizes

8.2 Histogram Equalization Results on LC2

(a) Original Dark Image (LC2)

(b) After Global Histogram Equalization

(c) Local Histogram Equalization (7×7)

(d) Local Histogram Equalization (31×31)

(e) Local Histogram Equalization (51×51)

(f) Local Histogram Equalization (71×71)

Figure 6: LC2 image with global and local histogram equalization using given window sizes

8.3 Comparisons of the results

Global histogram equalization overall enhances contrast across the entire image uniformly, often improving overall brightness but potentially losing local details in specific regions (e.g. darker background).

On the other hand, local histogram equalization adjusts contrast in smaller regions, resulting in better enhancement of fine details and textures, especially in areas with subtle variations. This enhances very fine details, but could also introduce noise or an over-sharpened effect, especially in regions with smooth transitions.

Overall the (7x7) size produces embossed image since the window size is so small it equalises the intensities in the image, preservnig edges but also adding noise in the image. The results of (31x31), (51x51) and (71x71) seem good-enough enhanced and comparable with GHE result.

8.3.1 LC1 Image

The Local Histogram equalization produces better enhancements than Global, specifically in regions, see (b) vs (d/e/f):

- windows of the house seem darker in GHE, and are enhanced in LHE
- the trees in the left background look well lit in LHE compared to the GHE one
- the floor in the overall seems well lit overall (foreground and even in darker background) and enhanced than GHE
- the branches of the front tree look highly enhanced, have high contrast with ackground and detailed in LHE than GHE

reason for the above being the variations in intensities in those regions in original image there is very less and hence not enhanced in Global method.

8.3.2 LC2 Image

Similar to the previous image, (7x7) looks embossed due to small window size, other than that, LHE results are better than GHE results in regions:

- the floor of forest is better enhanced, has good contrast and has more detailed visible shadows in LHE results than GHE result
- the trees in the background of GHE have low contrast and details compared to the LHE ones, where they are clear and have similar contrast as the foreground ones
- all the branches have good and equal contrast compared to GHE where there are variations in cotrast of branches depending on the position (may be more or less appealing depending on perspective, but still LHE is better details-wise)